**Use Case Title :** Weather Dashboard

**Student Name :** Sriram A

# Register Number : 20222141506124

**Institution :** Nazareth Margoschis College at Pillaiyanmanai

**Department :** Department of Computer Science

# Date of Submission : 22.03.2025

# 1. Problem Statement

Build a Weather Dashboard that fetches real-time weather data from the OpenWeather

API and displays it in a user-friendly web application.

# 2. Proposed Solution

The application should allow users to

1. Search for a city's weather details.
2. View temperature, humidity, and weather conditions.
3. Display weather icons based on the conditions (e.g., sun for clear weather, clouds for cloudy weather).

3. Technologies & Tools Considered

* HTML
* CSS
* Javascript

# 4. Solution Architecture & Workflow

The **Weather Dashboard** is a web-based application that allows users to search for weather details of any city using real-time data from the **OpenWeather API**.

**Major Components & Interactions**

# User Interface (Frontend)

* The user enters a city name in the search bar.
* Clicks the **Search** button to fetch weather data.
* Weather details (temperature, humidity, conditions) are displayed with icons.

# API Integration (Backend Communication)

* The system sends a request to the **OpenWeather API** with the city name.
* The API responds with **real-time weather data** in JSON format.

# Data Processing & Display

* The JavaScript code extracts relevant details from the API response.
* The extracted data is formatted and displayed dynamically on the webpage.

# Error Handling

 If the city is not found or there’s an issue with the API, an **error message** is displayed.



**User Inputs City**



**Clicks Search Button**



**Fetch Data from**

**OpenWeather API**



**API**

**Response**



**YES**

**NO**



**Extract Data**



**Show Error Message**



**Display Weather Info**

**5. Feasibility & Challenges**

**Feasibility**

This **Weather Dashboard** solution is highly **practical** due to the following reasons:

# Uses a Free and Reliable API

* The OpenWeather API provides real-time weather data, ensuring accuracy.
* It is free (with limits) and easy to integrate with simple HTTP requests.

# Lightweight & Fast

 The app uses **HTML, CSS, and JavaScript**, making it a **lightweight** solution.  No backend setup is needed, reducing complexity.

# User-Friendly Interface

* A simple UI with a search box and clear weather details improves usability.
* Weather icons provide **visual representation**, making it intuitive.

**Error Handling**

* Invalid city names show a **clear error message**, enhancing user experience.

# Responsive & Accessible

* The CSS ensures a **responsive design**, making it work on desktops and mobile devices.
* No heavy libraries or frameworks are required, making it **broadly compatible**.

**Challenges**

# API Limitations & Rate Limits

Free-tier APIs like OpenWeather have request limits, which might cause failure if too many users access the app simultaneously. **Solution:**

* Implement **caching** to store recent search results and reduce API calls.
* Use **local storage** or a small backend service to save previous searches.

**API Response Time & Network Issues** Slow or failed API responses due to network issues.**Solution:**

* Show aloading indicatorwhile fetching data.
* Usetry-catch blocksto handle API failures and suggest retrying.

# Security Concerns

API keys can be exposed in the frontend, leading to misuse.

**Solution:**

* Use **server-side proxy** to keep the API key hidden (e.g., a simple Node.js backend).
* Store API keys in **environment variables** when deploying to production.

# 6. Expected Outcome & Impact

This **Weather Dashboard** will provide several advantages by offering real-time weather updates in a simple, accessible, and efficient way.

**Key Benefits**

 Easy Access to Real-Time Weather Data

#  User-Friendly & Intuitive Interface  Fast & Lightweight Solution  Error Handling for Better User Experience

Users rely on full-scale weather websites or apps, which may be **slow, cluttered, or require logins**. With this **lightweight, real-time web app**, they get **instant results with minimal effort**.

Overall, this solution makes weather tracking simpler, faster, and more accessible to everyone. **Beneficial for Multiple Users**

|  |  |
| --- | --- |
| **General Users** | Quick weather updates for daily life. |
| **Travelers** | Helps plan trips based on current weather conditions. |
| **Event Organizers** | Assists in planning outdoor events. |
| **Farmers** | Provides insights on weather conditions for farming activities. |
| **Students & Researchers** | Useful for weather-related studies and projects. |

# 7. Future Enhancements

* **5-Day Weather Forecast** – Display a **multi-day forecast** instead of just current weather, helping users plan ahead.
* **Auto Location Detection** – Automatically fetch weather **based on the user’s current location**.
* **Dark Mode Toggle** – Add a **dark mode option** for better usability at night and to reduce eye strain.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**url:** [**https://sriramweatherapp.netlify.app/**](https://sriramweatherapp.netlify.app/)

**github:** [**https://github.com/Sriram-WebDev/Weather-Dashboard-**](https://github.com/Sriram-WebDev/Weather-Dashboard-)